SYSTEMS AND METHODS FOR ADAPTIVE INTERPRETATION OF INPUT FROM A TOUCH-SENSITIVE INPUT DEVICE

NOTICE OF COPYRIGHT PROTECTION

FIELD OF THE INVENTION

[0002] The present invention generally relates to receiving input from a touch-sensitive input device. This invention more particularly relates to adaptive interpretation of input received from a touch-sensitive input device.

BACKGROUND

[0003] A variety of input devices may be used to provide position and control data to programs executing on computers, cell phones, and other processor-equipped devices. These input devices include mice, trackballs, touchpads, touch screens, touch panels, and various other devices. While the mouse and trackball provide distinct control elements for performing positioning and other control actions, the touchpad combines positioning and control.

[0004] For example, a conventional mouse includes a ball or optical sensor for determining changes in position of the mouse. The mouse also includes one or more buttons for performing a control function, such as selecting a graphical representation on

a screen. In these systems, a user's intent to make a positional change or provide control input is apparent to the system.

[0005] In contrast, conventional touchpads combine the position and control functionality in a way that often masks the user's intent to make a positional change to provide control input. A user moves a finger along a touchpad to reposition a cursor. A user may also perform gestures to simulate functions of the buttons of a mouse, such as drag, click, and double-click. In either case, the user's finger is in contact with the surface of the touchpad. Changes in position on the touchpad and in the pressure exerted on the surface of the touchpad must be used to determine the user's intent. Because of the variety of users that may interact with a touchpad and the variety of functions that may be performed, determining the user's intent based on a gesture on a touchpad is difficult. Variables affecting the ability of a program to determine what a user is attempting to do include the following: the physical difference between users; the different angles at which a user may place their finger while using a touchpad; the variance in pressure between different users and between the same user; the movement of the finger across the touchpad while simultaneously attempting to perform actions on the touchpad. U.S. Patent Number 6,414,671 to Gillespie, *et al.* describes one conventional method for recognizing a user's gesture as a drag gesture.

[0006] Thus, a method and system are needed for accurately determining a user's intent based on data supplied by a touch-sensitive input device.

SUMMARY

[0007] An embodiment of the present invention provides systems and methods for adaptive interpretation of input received from a touch-sensitive input device by receiving a pressure signal indicating a pressure from the input device, comparing the pseudo pressure signal to an adaptive pressure threshold value, and outputting a signal if the pseudo pressure signal is greater than the adaptive pressure threshold value.

[0008] Further details and advantages of embodiments of the present invention are set forth below.

BRIEF DESCRIPTION OF THE FIGURES

[0009] These and other features, aspects, and advantages of the present invention are better understood when the following Detailed Description is read with reference to the accompanying drawings, wherein:

Figure 1 illustrates an exemplary environment for implementation of one embodiment of the present invention;

Figure 2 is a flow chart illustrating a process or algorithm for detecting finger presses on a touchpad in one embodiment of the present invention;

Figure 3 is a flow chart illustrating a process for detecting a finger press on a touchpad in another embodiment of the present invention; and

Figure 4 is a group of charts illustrating various filters that may be utilized in embodiments of the present invention.

DETAILED DESCRIPTION

[0010] Referring now to the drawings in which like numerals indicate like elements throughout the several figures, Figure 1 illustrates an exemplary environment for implementation of an embodiment of the present invention. The embodiment shown includes a touch-sensitive device commonly called a touchpad 102. Touchpad 102 senses the position of a conductor, such as a finger, on the surface of the touchpad (102). The touchpad (102) is further able to provide a position, comprising X and Y parameters, as well as a pressure, Z parameter, as an output signal. Conventional touchpads are very accurate in determining and providing the position of the conductor. For example, some conventional touchpads have resolutions greater than 1000 dpi. However, conventional touchpads are less accurate in determining and providing the pressure exerted on the touchpad. Other embodiments of the present invention may use other touch-sensitive input devices, such as a touch panel or touch screen.

[0011] The touchpad 102 shown does not sense an actual pressure. Instead, the pressure reading from the touchpad 102 is a pseudo pressure. Touchpads work by utilizing resistance, capacitance, or membrane switches. The touchpad 102 shown in Figure 1 utilizes capacitance, however, an embodiment of the present invention may be implemented in conjunction with any touch-sensitive input device, including resistive and membrane-switch touchpads. In other embodiments, actual pressure may be sensed. For example, in one embodiment, a touch screen with an attached explicit pressure sensor is utilized.

[0012] Capacitance-based touchpads are well known to those skilled in the art, and therefore, only a basic description of their function is provided herein. A capacitance touchpad, such as touchpad 102 shown in Figure 1, includes two sets of wires, which are perpendicular to one another and configured so that a gap is formed between them. When a user places a conductor, such as a finger, on the touchpad 102, wires of the two perpendicular sets are brought together and form a capacitance. The touchpad 102 measures which of the wires in each of the two sets has the most capacitance to determine where the conductor is touching the touchpad 102 and, based on this information, provides the X and Y coordinates of the position of the conductor on the touchpad 102.

[0013] The touchpad 102 also provides a pseudo pressure, Z. The pseudo pressure is based on the amount of capacitance resulting from the conductor touching the touchpad 102. Accordingly, the amount of capacitance is not a direct measure of pressure but rather a pseudo pressure.

[0014] In other words, the pseudo pressure or Z parameter provided by the touchpad 102 is not a measure of the actual vertical displacement by a conductor at a single point on the touchpad 102, but rather an estimation of the vertical displacement based on the size of the capacitance change. The pseudo pressure may not accurately represent the amount of pressure actually exerted on the touchpad 102. For example, the larger the surface of the conductor used on the touchpad 102, e.g., a user's finger, the larger the change in capacitance per amount of pressure exerted. As would be expected, if a user presses heavily against the touchpad 102 with a fleshy part of the finger, the amount of touchpad 102 area covered by the finger is greater than then when the same part of the

finger is touching lightly. However, what is less obvious is that the area covered, and the corresponding pseudo pressure, is also greater than when the user presses heavily with a bony part of a finger.

[0015] Additionally, the difference in the features of different conductors, for instance the size or makeup of different users' fingers, affects the capacitance change for any given change in pressure. For example, if a first user with a large finger applies the same pressure as a second user with a small finger, the pseudo pressure signal output by the touchpad 102 is greater for the first person than for the second person for the same amount of applied pressure.

[0016] The difficulty in determining a user's intent by evaluating the data provided by the touchpad 102 is compounded by the different ways in which a conductor may be utilized. For example, the pressure exerted across the surface of the touchpad may vary as the user's finger moves in relation to the hand. The user's finger covers a larger area of the touchpad when the finger is extended horizontally away from the hand on the touchpad 102 than when the finger is close to the hand. Similarly, a pointing device held vertical in relation to the touchpad 102 may cover a smaller surface area than one held at an angle to the touchpad 102.

[0017] Referring again to Figure 1, the touchpad 102 transmits the X, Y, and Z parameters 104 to a processor 106. The touchpad 102 in various embodiments of the present invention may be capable of sending several types of coordinate information. For example, a Synaptics TouchPad is able to send either relative or absolute coordinates. Relative coordinates provide the movement of the conductor on the touchpad 102 since

the last coordinates were transferred. Absolute coordinates provide the position of the conductor on the touchpad 102 at that moment. An embodiment of the present invention may utilize additional parameters as well. For example, the Synaptics TouchPad provides a "W" parameter, which reports the character of a contact with the touchpad, such as "accidental." An embodiment of the present invention may utilize such a parameter to accurately determine a user's intent.

[0018] Referring again to Figure 1, the processor 106 and touchpad 102 may be connected directly or indirectly and may be connected via wires or a wireless connection. For example, the touchpad 102 may utilize the PS/2, Serial, Apple Desktop Bus (ADB), or other communication protocol in communicating with the processor. The processor 106 is capable of executing program code stored on a computer-readable medium. Although the processor shown is separate from the touchpad 102, some conventional touchpads include a processor, such as an Application Specific Integrated Circuit (ASIC). An ASIC may provide some processing of the movements on the touchpad 102 to determine whether or not the user is making gestures. This integrated processor may be utilized alone or in combination with the processor 106 according to the present invention.

[0019] Processor 106 may include, for example, digital logic processors capable of processing input, executing algorithms, and generating output as necessary in response to the inputs received from the touch-sensitive input device. Such processors may include a microprocessor, the aforementioned ASIC, and state machines. Such processors include, or may be in communication with, media, for example computer-readable media, which

stores instructions that, when executed by the processor 106, cause the processor 106 to perform the steps described herein.

[0020] Embodiments of computer-readable media include, but are not limited to, an electronic, optical, magnetic, or other storage or transmission device capable of providing a processor, such as the processor 106 in communication with a touch-sensitive input device, with computer-readable instructions. Other examples of suitable media include, but are not limited to, a floppy disk, CD-ROM, magnetic disk, memory chip, ROM, RAM, an ASIC, a configured processor, all optical media, all magnetic tape or other magnetic media, or any other medium from which a computer processor can read instructions. Also, various other forms of computer-readable media may transmit or carry instructions to a computer, including a router, private or public network, or other transmission device or channel, both wired and wireless. The instructions may comprise code from any computer-programming language, including, for example, C, C#, Visual Basic, Java, and JavaScript.

[0021] The embodiment shown in Figure 1 may be implemented in a variety of devices. Such devices include personal computers, many of which include an integrated touchpad. Such devices may also include handheld devices, such as handheld organizers, cellular telephones, handheld communicators, MP3 players, GPS receivers, and the like.

[0022] Embodiments of the present invention may also be utilized to implement haptic effects in devices such as those mentioned above. In such an embodiment, the haptic effects result from various actions by a user interfacing with a touch-sensitive input device, and the effects may be based on the user's intent as determined by the

processor 106. Haptic effects may also result from interaction with software executing on a device in communication with the touch-sensitive input device.

[0023] Embodiments of the present invention address the difficulties faced in attempting to determine the intent of a user based on the X, Y, and Z parameters supplied by the touchpad 102. Examples of determining a user's intent include determining when a user is tapping or pressing on a specific portion of a touch-sensitive input device that corresponds to a control displayed on the input device or displayed on a separate, synchronized display.

[0024] Embodiments of the present invention provide systems and methods for adaptive interpretation of the intent of a user of a touch-sensitive input device. In one embodiment of the present invention, a processor receives a pressure signal indicating a pressure from the input device, compares the pressure signal to an adaptive pressure threshold value, and outputs a signal if the pressure signal is greater than the adaptive pressure threshold value. The pressure may be a pseudo pressure or an explicit pressure. Also, the pressure may be filtered.

[0025] Embodiments of the present invention may also utilize the velocity of the conductor across the touchpad in determining a user's intent. Additionally, an embodiment may utilize adaptive thresholds alone or in combination with digital filtering to more accurately determine a user's intent.

[0026] Thresholds for pressure, pseudo pressure, pseudo-pressure change, velocity, and other measures may be stored in a computer-readable medium when the device is

manufactured. Alternatively, software executed by a processor may provide settings for the thresholds. Thresholds set by software may be static or adaptive. Adaptive thresholds may rely on various parameters, including, for example, the length of time the input device has been active, the placement of the conductor on the surface of the input device, and the current user of the device.

[0027] Figure 2 is a flow chart illustrating a process or algorithm for detecting and interpreting finger presses on the touchpad (102) according to the present invention. In the embodiment shown, a keypad is displayed on the touchpad (102) or on a corresponding display. In various embodiments, the keypad may be virtual or physical, and may be displayed or not displayed. A processor executing the process shown compares the pseudo pressure against a minimum threshold value and compares changes in pseudo pressure against additional minimum thresholds.

[0028] The processor (106) may use adaptive thresholds. For example, the processor (106) may utilize different threshold values based on the position of the conductor on the touchpad (102). The processor (106) may also vary the thresholds based on the specific user who is touching the touchpad (102). The processor (106) may also vary the threshold when the user initially touches the touchpad (102) to account for the large change in pseudo pressure typically encountered during initial contact. For example, in one embodiment, the processor (106) varies the thresholds during the first one-half second that a pseudo pressure is detected, because the pseudo pressure value tends to vary drastically during the first one-half second of input. The variances may be based on activity of the user or upon the passage of time. The processor (106) may update the

threshold stored in memory, store a separate set of adaptive thresholds, or calculate and apply the adaptive thresholds on an ongoing basis.

[0029] In one embodiment in which an adaptive threshold is based on a specific user, the processor (106) executing the software is able to identify the user. A user identifier is stored in a computer-readable medium, and retrieved based on input received from a user, such as a user name, password, or other user identifier. Multiple user identifiers and threshold sets may be stored. The threshold may also depend on the orientation of the user's grip on a device used for pointing. For example, a stylus may incorporate a sensor to sense a user's grip orientation on the stylus.

[0030] Referring again to Figure 2, the processor (106) executes whenever the touchpad (102) is active 202. The touchpad (102) reports data continuously to the processor (106) at approximately 80 Hz. The processor (106) receives this data and uses it to determine the user's intent based on gestures made on the touchpad (102). The processor (106) first determines whether or not a finger or other conductor is on the touchpad (102) 204. The processor (106) determines that the finger is on the touchpad (102) by evaluating the pseudo-pressure (Z) parameter. If the Z parameter is greater than zero, the user's finger is touching. If not, the algorithm repeats step 204 until a finger is detected. If a finger is on the touchpad (102), the processor (106) determines whether a finger was previously on the touchpad (102) 206. The processor (106) may accomplish this in several ways. For example, the processor (106) may store the current or previous state of the touchpad (102) in memory and from that data, deduce whether the finger was previously on the touchpad (102).

[0031] If the finger was not previously on the touchpad (102), the processor (106) starts a first tick count 208. The first tick count is used to determine the length of time the finger remains on the key and is used in other parts of the algorithm for gesture recognition. If the finger was not on the touchpad (102) or after the first tick count is started, the processor (106) determines where the finger is positioned 210. The processor (106) makes this determination based on the X and Y coordinates provided by the touchpad (102).

[0032] In the embodiment shown, the processor (106) then utilizes the coordinates to determine whether the finger is on a key 212. Each key displayed on the touchpad (102) or corresponding display is associated with numerous attributes. These attributes include characteristics of the key, such as the size, position and behavior of the key. The processor (106) determines if the finger is on the key by comparing the X and Y position data reported by the touchpad (102) to the characteristics of the key. If the finger is not on a key, the processor (106) repeats the process beginning at step 204. If the finger is on a key, the processor (106) determines whether the release tick count has elapsed 214. If the release tick count has not elapsed, then the processor (106) repeats the process beginning at step 204. If the release tick count has elapsed, the processor (106) determines whether or not the first tick count has elapsed 216.

[0033] In the embodiment shown, if the first tick count has elapsed, the threshold is set to the move threshold for the key 218. If the first tick count has not elapsed, the threshold is set to the first threshold for the key 220. The threshold value is then compared to the change in pseudo pressure 222. If the change in pseudo pressure does

not exceed the threshold set in steps 218 and 220, the process repeats beginning at step 204. If the change in pseudo pressure exceeds the threshold value, the pseudo pressure, i.e., the current value of Z, is compared to an absolute threshold 224. If the pseudo pressure does not exceed the absolute threshold, the process repeats beginning at step 204. If the pseudo pressure exceeds the absolute threshold, then the processor (106) determines that the user is pressing the key 226. The processor (106) generates and sends a signal indicating that a press has been made. This signal is used by other software to control the flow of a program. For example, a word processing program may receive the signal, and in response, display a number, highlight a word, or perform some other action.

[0034] Once the determination that a press has occurred is made, the processor (106) starts the release tick count 228, and the process repeats beginning at step 204. As described above, the process continues to iterate for as long as the touchpad (102) is active.

[0035] In the process shown in Figure 2, the first tick count is set when the finger goes from a non-touching to a touching state and is used to measure a time interval during which a different (higher) set of thresholds is used because users typically push harder when they first touch a touchpad (102). The release tick count is used to measure a time interval following the detection of a press during which the finger is deemed to be pressing. During this interval, the processor (106) does not perform further press detection. In other words, the user cannot press again if the user is already pressing and the user cannot press any faster than some predetermined rate. Once the release tick

count expires, even if the user is still pressing hard, the algorithm detects a press if the user presses even harder (provided there is still room to press harder). The use of these tick counts provides for the adaptability of the algorithm.

[0036] Figure 3 is a flow chart illustrating another process according to the present invention for detecting a finger press on a touchpad (102). Similar to the embodiment shown in Figure 2, in the process shown in Figure 3, a processor (106) compares the pseudo pressure against a minimum threshold value and compares the change in pseudo pressure against a minimum threshold value. Also similar to the process above the thresholds may vary depending on where the finger touches the touchpad (102).

[0037] However, the process shown in Figure 3 differs from the process shown in Figure 2 in several ways. In the embodiment shown in Figure 2, the processor (106) compares the pseudo pressure against both lower and upper thresholds to determine whether the finger is touching. If the finger was not previously touching the touchpad (102), the processor (106) requires that the pseudo pressure exceed the upper threshold before the processor (106) can conclude that the finger is currently touching the touchpad (102). If the finger was previously touching the touchpad (102), the processor (106) requires that the pseudo pressure fall below the lower threshold before concluding that the finger is not touching the touchpad (102). Also, in the embodiment shown in Figure 3, the change in pseudo pressure is digitally filtered to reduce the effects of unwanted noise, which results from extraneous contact with the touchpad (102), such as sliding of a finger.

[0038] One digital filter useful in an embodiment of the present invention comprises software executing on a processor (106), such as a digital signal processor (DSP), to receive samples of data sent from a device, perform a numerical calculation on the data received, and provide the filtered data as output. The digital filter is programmable, allowing some signals to pass unaltered (passband) and blocks other signals (stopband). The signals between the passband and the stopband are signals in the transition band. A low-pass filter allows low-frequency (defined by the filter parameters) to pass. A high-pass filter allows high-frequency signals to pass. A band-pass filter allows frequencies to pass that are at some defined frequency, and a band-reject filter prevents certain signals from passing.

[0039] A recursive or non-recursive filter may be utilized in an embodiment of the present invention. A non-recursive or finite impulse response (FIR) filter utilizes only current input values for calculating an output value. A non-recursive filter does not use previous output values from the filter in computing the current output. In contrast, a recursive or infinite impulse response (IIR) filter utilizes both current input values and past output values in calculating the current output value. In one embodiment, the filter performs as a sliding window, placing more weight on recent values than on previous values.

[0040] A digital filter has order. The order of a non-recursive digital filter is equal to the number of previous input values used in the current calculation. The order of a recursive digital filter is the greater of either the number of (i) previous input values and (ii) previous output values that are used in the current output calculation. A non-

recursive filter can be a zero order filter. A recursive filter must by definition be at least a first order filter.

[0041] In the embodiment shown in Figure 3, the speed at which the finger is moving over the surface of the touchpad (102), i.e., the change in X and Y position on the touchpad (102) per cycle ($s = \sqrt{(\Delta X)^2 + (\Delta Y)^2}$), is also filtered and then compared against a maximum speed threshold. Until the speed falls below the maximum speed threshold, the processor (106) will not recognize a press.

[0042] In the embodiment shown in Figure 3, a processor (106) executing program code first compares the pseudo pressure to an upper threshold value 302. If the pseudo pressure exceeds the upper threshold value, the process continues at step 314. If not, the processor (106) determines whether the user was previously touching, for example by checking the value of a stored flag 304. If so, the processor (106) compares the pseudo pressure to a lower threshold value 306. If the user was not previously touching or if the pseudo pressure is equal to or below the lower threshold, the processor (106) determines whether the first tick counter has elapsed 308.

[0043] If the first tick counter has elapsed, the process resumes at step 302. If the tick counter has not elapsed, then the processor (106) concludes that the user is tapping 310. The processor (106) clears the first tick count, and the process returns to step 302.

[0044] At step 302, if the processor (106) determines that the pseudo pressure exceeds the upper threshold, the processor (106) determines whether the user was previously touching 314. If so, the processor (106) bypasses step 316 and concludes that

the user is touching 318. If the user was not previously touching at step 314, the processor (106) starts the first tick counter 316 and concludes that the user is touching 318. If the pseudo pressure is greater than the lower threshold 306 and the user was previously touching 304, the processor (106) restarts the tick count 316.

[0045] In any event, in the embodiment shown, once the processor (106) concludes that the user is touching 318, the processor (106) compares the speed to a speed threshold value 320. If the speed is greater than or equal to the speed threshold, the processor (106) returns to step 302 in the process. In comparing the speed to the speed threshold, the processor (106) may determine that although the user is exerting enough pressure to signify a press, because the finger is moving across the touchpad (102), the user does not intend for a press to be recognized.

[0046] If the speed is less than the speed threshold, the change in pseudo pressure is compared to a threshold value 322. If the change in pseudo pressure is less than or equal to the threshold, the processor (106) returns to step 302 in the process. If the change in pseudo pressure is greater than the threshold, the processor (106) determines whether the first interval has elapsed 324. If so, the processor (106) concludes that the user is pressing 326 and the processor (106) returns to step 302 in the process.

[0047] Embodiments of the invention may use filtering to reduce the effects of unwanted noise. In one such embodiment, three variables are filtered: (1) the speed at which the finger moves across the surface of the touchpad (102), (2) the pseudo-pressure (Z), and (3) the change in pseudo-pressure (Δ Z). The filtering of each of the above quantities may be performed using the same type of filter or different types of filters. For

example, in one embodiment, the quantities are filtered using a low-pass first-order recursive digital filter based on the following formula:

$$y(n) = \frac{x(n) + (N-1)y(n)}{N} \qquad \text{[Equation 1]}$$

[0048] N is a parameter affecting the cut-off frequency of the filter. For example, in one embodiment, N is set to 10 in filtering speed and pseudo pressure, and N is set to 5 in filtering the change in pseudo-pressure. These values are related to the sampling frequency of the touchpad (102), which, in the embodiment shown, is about 80 Hz. Such a filter computes a rolling average using a weighting function emphasizing more recent samples. This filter requires minimal computational and storage requirements.

[0049] In one embodiment, the following thresholds are used to detect finger presses:

Table 1:

Variable	**Threshold**
S	32
Z	*lower* = 16 *upper* = 32
ΔZ	User-dependent and location-dependent. Thresholds typically range from 2 to 8.

[0050] Ramp type filters work very well in filtering out unwanted noise. However, various types of filters, utilizing a variety of waveforms, a combination of filters, and other processing means may be used as part of a process for determining a user's intent. In one embodiment, the change in pseudo pressure is computed by subtracting the filtered

(average) pseudo pressure from the current pseudo pressure. In another embodiment, the previous filtered pseudo pressure is subtracted from the current filtered pseudo pressure.

[0051] In one embodiment, the user adjusts the threshold values for each key to attain maximum accuracy in intent determination. In another embodiment, the threshold is based on the standard deviation of the pseudo-pressure change. In yet another embodiment, more sophisticated filtering techniques are utilized. Figure 4 is a group of charts illustrating various filters that may be utilized in embodiments of the present invention. The following table provides the waveforms used for each filter shown:

Table 2:

Reference	Waveform	Formula
402	Step	$\lambda(n) = \begin{cases} 1, & n < N/2 \\ 0, & n = N/2 \\ -1, & n > N/2 \end{cases}$
404	Pulse	$\lambda(n) = \begin{cases} -1, & n < N/4, n > 3N/4 \\ 0, & n = N/4, n = 3N/4 \\ 1, & N/4 < n < 3N/4 \end{cases}$
406	Ramp	$\lambda(n) = 1 - 2n/N$
408	Triangle	$\lambda(n) = \begin{cases} 4n/N - 1, & n \le N/2 \\ 3 - 4n/N, & n > N/2 \end{cases}$
410	Quarter Cosine	$\lambda(n) = -\sin(\pi n/2N)$
412	Quarter Sine	$\lambda(n) = \cos(\pi n/2N)$
414	Half Cosine	$\lambda(n) = \cos(\pi n/N)$
416	Half Sine	$\lambda(n) = \sin(\pi n/N)$
418	Full Cosine	$\lambda(n) = -\cos(2\pi n/N)$

[0052] In various embodiments, the coefficients of these waveforms are further biased so their average is zero and scaled so the sum of the positive coefficients is one.

More formally, the filter coefficients $a(n)$ are computed from the above coefficients $\lambda(n)$ by the following equations:

$$\beta = \frac{1}{N+1}\sum \lambda(n) \qquad \text{[Equation 2]}$$

$$\nu(n) = \lambda(n) - \beta \qquad \text{[Equation 3]}$$

$$\rho(n) = \begin{cases} \nu(n), & \nu(n) > 0 \\ 0, & \nu(n) \le 0 \end{cases} \qquad \text{[Equation 4]}$$

$$\mu = \sum \rho(n) \qquad \text{[Equation 5]}$$

$$a(n) = \frac{\rho(n)}{\mu} \qquad \text{[Equation 6]}$$

where:

B	is the bias among $\lambda(n)$,
$N(n)$	are the unbiased coefficients,
$P(n)$	are the positive coefficients extracted from $\nu(n)$,
M	is the sum of the positive coefficients, and
$A(n)$	are the final filter coefficients.

[0053] The foregoing description of the preferred embodiments of the invention has been presented only for the purpose of illustration and description and is not intended to be exhaustive or to limit the invention to the precise forms disclosed. Numerous modifications and adaptations thereof will be apparent to those skilled in the art without departing from the spirit and scope of the present invention.